Attorney Matter No. 63564-071
Attorney Docket No. ACCL-132

UNITED STATES PATENT APPLICATION

FOR

GENERATION OF RECONSTRUCTED IMAGES

Assignee: Accuray, Inc.
Inventor: Gopinath Kuduvalli

McDermott, Will & Emery
28 State Street
Boston, MA 02109

GENERATION OF RECONSTRUCTED IMAGES

BACKGROUND

[0001] The need for computing DRRs (digitally reconstructed radiographs) arises in a number of situations. In image-guided surgery and radio-surgery, for example, DRRs from CT (computed tomography) scans of a patient are correlated with live patient images in order to determine the patient displacement from the desired position for administering the surgical treatment. Typically, the generation of a DRR for each CT orientation involves a large number of computations.

[0002] Speed is of essence when the DRRs are computed in real time, for example in order to determine the patient position, and in order to dynamically correct for patient displacement during the course of the surgical or radiosurgical treatment. Even when a library of DRRs is generated offline in order to choose a best match for the live images from the DRRs, fast DRR generation will allow for higher accuracy by generating an adequate number of DRRs to achieve the desired accuracy for determining the patient position.

[0003] Several algorithms for achieving fast DRR generation have been proposed. For example, in one known method an intermediate representation of the attenuations through the 3D CT volume, called a transgraph, is used to speed up the computation of DRRs. In the transgraph method, only a limited number of the transgraphs are computed upfront, however, so that the resulting DRRs are approximations for most positions of interest. Further, the range of orientations for which the DRRs can be computed is severely limited.

[0004] There is a need for a method and system for rapidly generating DRRs (or other forms of reconstructed 2D images) from 3D scan data.

SUMMARY

[0005] A method is presented for generating a DRR of a 3D scan volume f(x,y,z) of an object, from 3D scan data representative of the 3D scan volume, for a desired orientation (θ, ϕ, φ) of the 3D scan volume f(x,y,z). The method includes computing in frequency space the 3D Fourier transform F(u,v,w) of the 3D scan volume f(x,y,z). The values of F(u,v,w) are sampled along a surface S(θ, ϕ, φ, u',v') within the 3D data set representative of the Fourier transform F(u,v,w). For parallel beam geometry, the surface S(θ, ϕ, φ, u',v') is a plane. The 2D inverse Fourier transform F^{-1}[S(θ, ϕ, φ, u', v')] of the sampled surface S(θ, ϕ, φ, u', v') is computed, to obtain a DRR along a projection direction perpendicular to the surface S(θ, ϕ, φ, u',v').

[0006] The method includes a procedure for handling the cone-beam projections required for most realistic imaging applications. In one embodiment, cone-beam projections are handled by choosing the sampling surface S(θ, ϕ, φ, u',v') so that the sampled surface is part of the surface of a sphere, where the center of the sphere coincides with the origin of the projection x-rays. The 2D inverse Fourier transform F^{-1}[S(θ, ϕ, φ, u', v')] of the sampled surface S(θ, ϕ, φ, u', v') is computed, thereby obtaining a DRR along a projection that originates at the center the sphere, and projects on to a 2-D surface.

[0007] A system is presented for generating a DRR of a 3D scan volume f(x,y,z) of an object, for any desired orientation (θ, ϕ, φ) of the 3D scan volume, from 3D scan data representative of the volume f(x,y,z). The system includes a controller having an input module for receiving the 3D scan data. The controller includes a first processor configured to compute a 3D data set in frequency space representative of the Fourier transform F(u,v,w) of the 3D scan volume f(x,y,z). The controller further includes resampling software for resampling the 3D data set along a surface S(θ, ϕ, φ, u', v'). The surface S(θ, ϕ, φ, u', v') passes through the origin of the 3D F(u,v,w) data set, and is defined at angles (θ, ϕ, φ) corresponding to the desired orientation (θ, ϕ, φ) of the 3D scan volume. The controller includes a second processor configured to compute a 2D inverse

Fourier transform F^{-1} [$S(\theta, \phi, \varphi, u', v')$] of the surface $S(\theta, \phi, \varphi, u', v')$. The 2D inverse transform is a DRR along a projection direction perpendicular to the surface.

BRIEF DESCRIPTION OF THE DRAWINGS

[0008] FIG. 1 schematically illustrates the generation of 2D reconstructed images of an object from 3D scan data of the object, as known in the art.

[0009] FIG. 2 is an illustration of the 2D Fourier slice theorem.

[0010] FIG. 3 schematically illustrates the extension into three dimensions of the 2D Fourier slice theorem.

[0011] FIG. 4 is a schematic flow chart of a method for fast generation of 2D DRRs using 3D fast Fourier transform.

[0012] FIG.s 5A – 5C illustrate the padding of the 2D sample surface with zeros.

[0013] FIG. 6 is a schematic block diagram of a system for fast generation of 2D reconstructed images.

[0014] FIG. 7 provides a table that compares the number of computations required by methods known in the art, with the number of computations required by DRR generation using 3D fast Fourier transform.

DETAILED DESCRIPTION

[0015] A method and system is presented for fast generation of DRRs from a 3D scan volume, for any desired orientation (θ, ϕ, φ) of the 3D scan volume. A 3D fast Fourier transform of the 3D scan volume is computed. The 3D Fourier transform data are sampled along a surface perpendicular to the desired direction of projection of the DRR. The inverse Fourier transform of the sampled surface is computed, to generate a DRR for the desired orientation (θ, ϕ, φ) of the 3D scan.

[0016] FIG. 1 schematically illustrates the generation of 2D reconstructed images of an object from 3D scan data of the object, as known in the art. In the particular illustrated embodiment, the 2D reconstructed images are DRRs (digitally reconstructed radiographs), and the 3D scan data are CT scan data. In other embodiments of the invention, however, other types of 3D scan data may be used, including but not limited to, MRI (magnetic resonance imaging) scan data, PET (positron emission tomography) scan data, and ultrasound scan data), and reconstructed images other than DRRs may be generated.

[0017] In FIG. 1, the volumetric 3D CT image of the object is referred to with the aid of reference numeral 20. The DRRs 25A and 25B, shown in FIG. 1, are artificial, synthesized 2D images that represent the radiographic image of the object that would be obtained, if imaging beams having a hypothetical intensity, position and angle were used, and if the object were positioned and oriented in accordance with the 3D CT scan data. In other words, the DRRs are calculated from 3D CT data, in an exact emulation of hypothetical or known camera perspectives. The reference numerals 30A and 30B illustrate the hypothetical positions and angles from which the imaging beams would be directed, through an object positioned in accordance with the CT volumetric image 20 of the object.

[0018] It is known to generate DRRs by casting hypothetical beams or rays through the CT volumetric image 20 of the target. Each hypothetical ray passes through a number of voxels of the 3D CT image 20. By integrating the CT numbers for these voxels along

each ray, and projecting onto an imaging plane (shown as 70A and 70B, respectively, in FIG. 1), the resultant image would emulate the radiograph that would be obtained by passing rays from hypothetical camera locations and angles (shown schematically as 30A and 30B, respectively) through an object positioned in accordance with the volumetric 3D image 20. Ray tracing algorithms, known in the art, are generally used to generate the DRRs.

[0019] The method and system described in this patent make use of the well known Fourier slice theorem, in order to reduce the number of computations required to generate a DRR for a desired orientation of the CT image 20, and thereby increase the speed of DRR generation. The Fourier slice theorem states that the Fourier transform of a parallel projection of an object is equal to a slice out of the Fourier transform of the object itself. Considering a particular function f(x, y), which represents e.g. a 2D image of an object in a 2-dimensional x-y Cartesian coordinate system, the one-dimensional Fourier transform of the parallel projection of f(x,y), at some angle θ, has the same value as a line (or "slice") of the 2D Fourier transform of f(x,y) itself at that same angle θ, according to the Fourier slice theorem.

[0020] It is known in the art to use the Fourier slice theorem to estimate the image function f(x,y) using the projection data: by collecting the one-dimensional Fourier transforms of the projections at enough projection angles, an estimate of the two-dimensional transform can be made, which can then be simply inverted to estimate the image function f(x,y). In contrast these known methods, in this patent the Fourier slice theorem is extended to three dimensions. The 3D Fourier slice theorem is then used to more rapidly generate 2D projection images from the available 3D scan data, as described below.

[0021] FIG. 2 is a graphical illustration of the 2D Fourier slice theorem, known in the art. FIG. 2 shows that the 1D Fourier transform of a projection (at angle θ), is a line in the 2D Fourier transform of the 2D function f(x,y) from which the projection is computed, where the normal to this line represents the projection angle θ. The projection $P(\theta,t)$ of

the function f(x,y) at an angle θ is the summation of pixels of the image f(x,y) along lines normal to an axis t, as illustrated in FIG. 2. As seen from FIG. 2, the axis t can be represented, in terms of the x-y coordinates, as follows:

$$t = x \cos \theta + y \sin \theta. \quad (1)$$

[0022] An axis n (shown in FIG. 2), perpendicular to the axis t, and passing through the origin, can be defined in the terms of x-y coordinates, as follows:

$$n = - x \sin \theta + y \cos \theta. \quad (2)$$

Since the projection $P(\theta,t)$ is a summation of pixels of f(x,y) along lines that make an angle θ + 90 degrees with the x-axis, it may be written in terms of an integral over the n-axis:

$$P(\theta,t) = \int f(x,y)\, dn. \quad (3)$$

From equations (1) and (2), it can readily be seen that

$$x = t \cos \theta - n \sin \theta \quad (4)$$

$$y = t \sin \theta + n \cos \theta. \quad (5)$$

Substituting (4) and (5) into (3), the projection $P(\theta,t)$ is given by:

$$P(\theta,t) = \int_{-\infty}^{+\infty} f(x,y)\, dn = \int_{-\infty}^{+\infty} f(t \cos \theta - n \sin \theta, t \sin \theta + n \cos \theta)\, dn. \quad (6)$$

[0023] As well known, the Fourier transform of a function represents a function as a summation (or integral) of a series of sine and cosine terms of increasing frequency, and describes the amount of each frequency term that must be added together to form the function. The mathematical expression for a 1D Fourier transform F_u of f(t) is given by:

$$F_u [f(t)] = \int_{-\infty}^{+\infty} f(t)\, e^{-2\pi j u t}\, dt, \quad (7)$$

and the mathematical expression for a 2D Fourier transform $F_{(u,v)}$ of f(x,y) is given by:

$$F_{(u,v)} [f(x,y)] = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x,y)\, e^{-2\pi j (ux+vy)} dx\, dy. \quad (8)$$

Using equation (7), one can take a one-dimensional Fourier transform of both sides of equation (6) above:

$$F_u [P(\theta,t)] = \int_{-\infty}^{+\infty} [\int_{-\infty}^{+\infty} f(t \cos \theta - n \sin \theta, t \sin \theta + n \cos \theta)\, dn\,] e^{-2\pi j u t}\, dt.$$

Using equations (1), (4) and (5), the above expression becomes:

$F_u [P(\theta,t)] = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x,y)\, e^{-2\pi j u(x \cos\theta + y \sin\theta)}\, dn\, dt.$

Since $\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} dn\, dt = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} dx\, dy$, it follows that:

$F_u [P(\theta,t)] = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x,y)\, e^{-2\pi j[(u \cos\theta) x + (u \sin\theta) y]}\, dx\, dy.$

Using equation (8), we finally obtain:

$$F_u [P(\theta,t)] = F_{(u \cos\theta,\, u \sin\theta)} [f(x,y)] \qquad (9)$$

[0024] Equation (9) is a mathematical expression of the two-dimensional Fourier slice theorem. The left hand side of Eq. (9) is the one dimensional Fourier transform of a projection of f(x,y) along lines that make an angle of $\theta + 90°$ with the x-axis. The right hand side is the two dimensional Fourier transform of f(x,y) along a radial line in frequency space, the radial line making an angle θ and passing through the origin in frequency space. FIG. 2 graphically illustrates Eq. (9), showing that the 1D Fourier transform of a projection vector $P(\theta,t)$ of f(x,y), along an axis t that makes an angle θ with respect to the x- axis in spatial dimensions, is a radial line in the 2D Fourier transform F(u,v) of the image function, the radial passing through the origin in frequency space, and making an angle θ with the horizontal axis.

[0025] It follows from the 2D Fourier slice theorem that the projection of f(x,y) along lines normal to an axis t can be generated by simply taking the inverse Fourier transform of the radial line (or "slice") of the 2D Fourier transform of the original data set f(x,y). The resulting complexity of the computations required to generate such a projection is only of the order of O(Nlog N), assuming that f(x,y) is represented by an array of data points having dimensions NxN), instead of $O(N^2)$.

[0026] In one embodiment, the above-described property of the Fourier slice theorem is used in order to reduce the number of computations required for generating 2D DRRs from 3D scans, by using an extension of the 2D Fourier slice theorem into three dimensions, i.e. the 3D Fourier slice theorem. Considering a 2D parallel projection of a 3D object, where the normal to the 2D projection is parallel to the projection direction, the 3D Fourier slice theorem states that the 2D Fourier transform of the 2D projection

corresponds to a planar slice through the 3D Fourier transform of the object, the planar slice passing through the origin and having a normal parallel to the projection direction.

[0027] FIG. 3 schematically illustrates the extension into three dimensions of the 2D Fourier slice theorem. Referring to FIG. 3, the 2D projection of the 3D image volume f(x,y,z) onto a surface defined by angles (θ, ϕ, φ) is a 2D summation of pixels in the 3D image volume f(x,y,z), along lines perpendicular to the surface at angles (θ,ϕ,φ), by analogy to the two dimensional case. The representation in FIG. 3 of the surface of projection is given using a normal vector n perpendicular to the projection surface. For consistency in comparing with FIG. 2, F(u,v,w) denotes the 3D Fourier transform of the 3D image volume f(x,y,z), and F[P(θ, ϕ, φ, r, t)] denotes the 2D Fourier Transform of the 2D projection P(θ, ϕ, φ, r, t). Then the 3D Fourier slice theorem can be stated as follows: "the 2D Fourier Transform F[P(θ, ϕ, φ, r, t)] of the projection image $P(\theta(\phi,\varphi,r,t)$ along the axes (r,t), is a surface at angles (θ, ϕ, ϕ) in the 3D Fourier transform F(u,v,w) of the image f(x,y,z)." The 2D surface passes through the origin of the 3D Fourier transform, and has the same orientation (θ, ϕ, ϕ).

[0028] Using the 3D Fourier slice theorem, projections of a 3D CT scan volume f(x,y,z) at arbitrary angles can be computed quickly, by first computing the 3D forward Fourier transform of the 3D function f(x,y,z) as a preprocessing operation, then taking 2D slices out of that 3D transform, and performing inverse 2D transforms of the 2D slices. Computation time is reduced, once the initial one-time step of computing the Fourier transform has been taken care of, because the computation intensive 3D Fourier transform needs only be computed once, in the beginning. Once this step is completed, only 2D manifolds of data need to be handled, thereafter, thereby considerably increasing the efficiency of generating DRRs.

[0029] FIG. 4 is a schematic flow chart 100 of a method for fast generation of 2D DRRs, in accordance with one embodiment. Generation of fast DRRs using this method consists of finding the appropriate surface within the 3D Fourier transform that matches the 2D Fourier transform of the observed 2D DRR image representing the projection of

the 3D object from some orientation. By choosing surfaces at any desired orientation, and computing the inverse Fourier transform of that surface, DRRs for any desired orientation of the CT scan volume can be rapidly computed.

[0030] In overview, the method for fast generation of DRRs consists of steps 110, 120, and 130, as schematically illustrated in FIG. 4. In step 110, the 3D Fourier transform F(u,v,w) of the function f(x,y,z) is computed, where f(x,y,z) represents the 3D scan volume. This is the initial preprocessing 3D transformation step. After this step, two operations are performed in order to generate a 2D DRR at some desired projection angle. First, in step 120, a surface S (which is a surface for parallel beam geometry, and which is part of a spherical surface for cone beam geometry) is extracted out of the 3D data set representing F(u,v,w) that is perpendicular to the direction of projection of the DRR, i.e. an appropriate matching surface is chosen from the 3D F(u,v,w) data set. This step involves resampling in the frequency domain. The 3D array of F(u,v,w) data is resampled along a surface. Next, in step 130, the 2D inverse Fourier transform of the extracted surface is calculated. This 2D inverse Fourier transform is a DRR along a projection direction perpendicular to the extracted surface.

[0031] Once step 110 of computing the transform F(u,v,w) has been performed in the beginning, steps 120 (resampling along a surface) and 130 (taking the inverse 2D Fourier transform of the resampled surface) can be performed for any desired direction of projection, i.e. for any desired orientation of the 3D scan volume. DRRs for different viewing angles can be obtained simply by repeating steps 2) and 3). In this way, DRRs can be rapidly generated, at any desired projection direction, i.e. for any desired orientation of the 3D scan volume. Subsequent to step 1), the required complexity of computations is of the order of $O(N^2 \log N)$, which is the complexity of the inverse Fourier transform operation, where the dimensions of the 3D data set for F(u,v,w) is given by N*N*N.

[0032] The resampling step 120 is a process that involves the extraction and interpolation of grey levels from pixel locations in the original 3D data set. Resampling

involves: 1) positioning the origin of the resampling kernel at the data point to be resampled; 2) multiplying neighboring pixel values with the resampling kernel value at those data points; and 3) adding together the multiplied values, to form the resampled value. To optimize the resampling process, an appropriate 3D resampling kernel should be selected for sampling the values across the surface $S(\theta, \phi, \varphi, u', v')$. A number of resampling methods may be used, including but not limited to: nearest neighbor resampling method; bi-linear interpolation; tri-linear interpolation; sinc function (sin(x) / x) resampling kernel method. Other resampling methods known in the art can also be used, including but not limited to: resampling using other types of resampling kernels such as the Gaussian resampling kernel or the spline resampling kernel; and cubic convolution.

[0033] The resampling method that is most efficient, in terms of computation time, is the nearest neighbor resampling method. Nearest neighbor interpolation determines the grey level from the closest pixel to the specified input data coordinates, and assigns that value to the output data coordinates. In other words, nearest neighbor resampling copies the value from the nearest reference pixel to that location.

[0034] For better accuracy, bi-linear, tri-linear, or bi-sinc kernels can be used. Bi-linear interpolation determines the grey level from the weighted average of the four closest pixels to the specified input data coordinates, and assigns that value to the output data coordinates. Tri-linear interpolation is an improvement on linear interpolation, and ramps between several different linear interpolations. Two pixels that are closest to the input coordinates are picked, and two weights are computed that are applied to the pixel values for each of the two pixels. For each of those two pixel values, the four closest pixels at each value are found, and bi-linear interpolation is performed between these four pixels to find the pixel value at that position. The two pixel values are then combined using appropriate weights, to generate the final interpolated value.

[0035] A number of additional optimizations can be performed, when sampling the surface $S(\theta, \phi, \varphi, u', v')$ from the 3D fast Fourier transform volume. These

optimizations include: 1) selecting only a sub-volume around the origin to resample; 2) padding the 2D sample surface with zeros; and 3) applying a convolution filter, by multiplying the Fourier sample surface with a 2D Fourier transform of the convolution filter. Since a 3D fast Fourier transform concentrates most of the information close to the origin, efficiency can be achieved by selecting only a sub-volume around the origin to resample.

[0036] Resampling in the frequency domain can introduce aliasing artifacts. To prevent aliasing, the 3D F(u,v,w) data set must be sampled at a high enough rate. Zero-padding may be require in order to generate a DRR of appropriate pixel spacing. FIG.s 5A, 5B and 5C schematically illustrate the padding of the 2D sample surface with zeros. FIG.s 5A and 5B illustrate the 2D projections of F(u,v,w), in the z-x surface and the z-y surface, respectively. In both figures, a padding of size n are used on the top and bottom of the F(u,v,w) volume, along the z-axis. FIG. 5C provides a 3D illustration of zero-padding.

[0037] For parallel beam geometry, the surface $S(\theta, \phi, \varphi, u',v')$ is a plane. For cone-beam geometry, the surface $S(\theta, \phi, \varphi, u',v')$ is part of a sphere whose center is coincident with the imaginary origin of the x-rays for projection. In one embodiment, the resampling step 120 includes a procedure for handling cone-beam projections, which are required for most realistic imaging applications. Cone-beam projections refer to the projection geometry closely approximated by real world imaging systems, in which the x-rays (or other type of imaging beams) originate from a single point source, pass through the object of interest, and are incident upon a 2D flat surface such as a 2D x-ray detector array. In order to handle cone-beam projections, the sampling surface $S(\theta, \phi, \varphi, u',v')$ is chosen in such a way that the surface is part of the surface of a sphere, where the center of the sphere coincides with the point of origination of the x-rays or other imaging beam. The 2D inverse Fourier transform $F^{-1}[S(\theta, \phi, \varphi, u', v')]$ of the sampled surface $S(\theta, \phi, \varphi, u', v')$ is then computed. In this way, a DRR is obtained along the direction of a projection that originates at the center the sphere, and projects

on to a 2-D surface.

[0038] FIG. 6 is a schematic block diagram of a system 200 for fast generation of 2D reconstructed images of an object using 3D fast Fourier transforms. In the illustrated embodiment, the system 200 includes a 3D scanner 210 (including but not limited to a CT scanner, a PET scanner, an MRI scanner, and an ultrasound scanner) that generates 3D scan data representative of an object. In particular, the 3D scan data includes scan data representative of a 3D scan volume (represented as f(x,y,z) in an x-y-z Cartesian coordinate system) of the object, the 3D scan volume having a certain 3D orientation defined by angles (θ, ϕ, φ). In other embodiments, however, the system 200 does not include a 3D scanner, and simply receives 3D scan data representative of f(x,y,z) that have been generated elsewhere.

[0039] The system 200 includes a controller 220. The controller 220 includes an input module 225, or other means for receiving the 3D scan data of f(x,y,z), which may be provided by the scanner 210, or which may have been generated elsewhere. The controller 220 includes a first processor 230 that includes 3D Fourier transform software and that is configured to compute a 3D data set representative of a Fourier transform F(u,v,w) of the 3D scan volume f(x,y,z). The variables u, v, w respectively represent variables along three mutually orthogonal coordinate axes in the frequency domain. Computation of 3D Fourier transforms is well known, and standard software and/or algorithms that are commercially available may be used.

[0040] The controller 220 includes resampling software 240 for resampling the 3D F(u,v,w) data set along a surface S(θ, ϕ, φ, u', v') that passes through the origin of the 3D F(u,v,w) data set, and that is defined at angles (θ, ϕ, φ) corresponding to the orientation of the 3D scan volume f(x,y,z). In one embodiment, the resampling software 240 includes software for performing nearest-neighbor resampling. In this embodiment, the resampling software assigns, to each pixel along the surface S(θ, ϕ, φ, u', v') in the 3D F(u,v,w) data set, the value of the closest neighboring pixel. In other embodiments, the resampling software 240 includes software for performing resampling methods that

involve an appropriate resampling kernel. In these embodiments, the resampling software multiplies one or more neighboring pixel values with the resampling kernel value at a sample data point, for each data point along the surface each data point along the surface $S(\theta, \phi, \varphi, u', v')$. The resampling software then adds together the multiplied values to form the resampled pixel value. Possible resampling kernels that can be used include, but are not limited to: a bi-linear resampling kernel; a tri-linear resampling kernel; a sinc resampling kernel; and a bi-sinc resampling kernel. Resampling 3D data sets is also well known in the art, and standard commercially available software and/or algorithms may be used.

[0041] In a particular embodiment, the resampling software 240 includes software for padding the sample surface $S(\theta, \phi, \varphi, u', v')$ with zeros, to reduce aliasing artifacts. In another embodiment, the resampling software 240 includes software for computing a 2D Fourier transform of a convolution filter, and for multiplying the surface $S(\theta, \phi, \varphi, u', v')$ with the 2D Fourier transform of the convolution filter.

[0042] The controller 220 includes a second processor 250 that includes inverse Fourier transform software, and that is configured to compute the 2D inverse fast Fourier transform $F^{-1}[S(\theta, \phi, \varphi, u',v')]$ of the resampled surface $S(\theta, \phi, \varphi, u', v')$. The inverse transform $F^{-1}[S(\theta, \phi, \varphi, u',v')]$ is a DRR along a projection direction perpendicular to the surface $S(\theta, \phi, \varphi, u', v')$. Computation of inverse Fourier transforms is also well known, and standard software and/or algorithms that are commercially available may be used.

[0043] The DRR generation technique described above is about an order of magnitude faster than existing methods of DRR generation of equivalent accuracy for offline DRR generation. After accounting for an initial cost for computing the 3D Fourier transform, the technique is faster than existing methods by about two orders of magnitude, for real-time DRR generation. The DRR generation technique described has a computational complexity proportional to *MxN*, compared to the complexity of traditional methods of *MxNxP*, where *M* and N respectively represent the number of rows and columns of the DRRs, and *P* represents the number of layers in the CT volume.

[0044] FIG. 7 provides a table ("Table 1") that compares the number of computations required by existing known methods, with the number of computations required by a DRR generation technique using 3D fast Fourier transform, and the 3D Fourier slice theorem, as described above. Typically, current known methods for generating a DRR having a size of *MxN* from a CT scan data volume of *MxNxP*, using a resampling kernel of size k, requires *M*N*P*k* computations.

[0045] In contrast, the computational complexity of the DRR generation technique described above is considerably reduced, for the following reasons. To resample a plane of size *MxN* from the 3D fast Fourier transform volume requires *M*N*k* computations. To compute the DRRs from this plane using inverse fast Fourier transforms requires $M*N*(log_2M+log_2N)$ computations. Therefore, the total number of computations, per DRR, is $M*N*(k+log_2M+log_2N)$. The additional one-time overhead of computing the 3D fast Fourier transform of the CT volume involves $M*N*P(log_2M+log_2N+log_2P)$ computations.

[0046] Table 1 in FIG. 7 gives a comparison of the computational complexity for CT volume sizes of 32 to 512 pixels in all three dimensions. The dimensions of the DRR are assumed to be same as the dimensions of the CT, for convenience and simplicity of illustration, although of course this restriction is not necessary for implementing the method and system described above, and other embodiments may include DRRs and CT scans having arbitrary dimensions. For each method, two numbers are computed: the first column in Table 1 gives the number of computations for each DRR. When computing this number, the overhead involved in computing the 3D fast Fourier transform has been ignored. Therefore, this column is useful for comparing the computational complexity involved in real-time DRR generation. The second column in Table 1, under each listed algorithm, gives the number of computations for computing 100 DRRs. The number in the second column includes the overhead involved in initially computing the 3D fast Fourier transform. The second column in Table 1 is thus useful for comparing the performance of the fast DRR generation technique (describe above) for computing a library of DRRs.

[0047] From the table in FIG. 7, it can be observed that for real-time DRR generation, the DRR generation technique described above is more than 25 to 500 times faster than the existing methods. For off-line DRR generation, the technique is more than 10 to 20 times faster than the existing algorithms.

[0048] In sum, a method and system for fast DRR generation, using an extension of the 2-D Fourier slice theorem to three dimensions, has been described. This technique is one or more orders of magnitude (11 to 25 times) faster than the current methods that operate in the 3-D space domain. This technique is about two or more orders of magnitude (25 to 500 times) faster for real-time DRR generation. Further, this technique provides additional processing advantages over existing methods. For example, a smoothing filter can be achieved by zeroing the higher order frequency components in the spatial frequency domain. Also, an appropriate convolution filter can be applied efficiently in the spatial frequency domain, by multiplying the 2D sample surface in the frequency domain by the 2D fast Fourier transform of the convolution filter. Cone beam projections, required for most realistic imaging applications, can also be handled.